# SQUALE – Software QUALity Enhancement

Alexandre Bergel[1] , Simon Denier[1], Stéphane Ducasse[1], Jannik Laval[1],
Fabrice Bellingard[2], Philippe Vaillergues[2], Françoise Balmas[3],
Karine Mordal-Manet[3]
[1] RMoD Team, INRIA, Lille, France
[2] Qualixo, Paris, France
[3] Labo IA, University of Paris 8, France
http://www.squale.org

The Squale project was born from industrial effort to control software quality. Its goals are to refine and enhance Qualixo Model, a software-metric based quality model already used by large companies in France (Air France-KLM, PSA Peugeot-Citroën) and to support the estimation of return on investment produced by software quality. Qualixo Model is a software quality model based on the aggregation of software metrics into higher level indicators called practices, criterias and factors. The coordination of Squale is carried out by Qualixo[1].

## 1 Funding and Global Effort

Squale stands for Software QUALity Enhancement. It is a national project, supported and labeled by the "Systematic - PARIS Region" competitive Cluster, and partially funded by Paris region and the DGE ("Direction Générale des Entreprises") in the context of the French Inter-ministerial R&D project 2006 - 2008 ("Projet R&D du Fonds Unique Interministériel"). The total cost of the project is 3.1 million Euros for a two year duration. The project started administratively in June 2008, and concretely in September 2008. The total effort on the project is about 25 person-years.

[1]http://www.qualixo.com

## 2 Project Goals

Over the last decade, the need for quality in software has increased. Several quality models have been proposed [1, 7, 8, 4, 5, 3]. These models emphasize the need to have quality checks while developing a software program. As far as we are aware of, no model to assess quality of existing software programs have reached a significant acceptance. The project stems from the experience gained by the SME Qualixo with the assessment of software quality for PSA, Air France-KLM and other large companies. The main goals of Squale are to evaluate and enhance the existing software quality approach in the following areas:

- evaluate and enhance the current software-metric quality model, including current practices and current metrics,
- define dashboards related to software quality,
- enabe assessment evolution of software quality,
- provide economical indicators to assess added value (ROI) of software quality measurements, and
- disseminate acquired knowledge through an open-source platform supporting the model and a community of users.

Finally the Squale project aims at building an anonymous centralized database where audit results will be stored in order to:

- offer the possibility - for Squale users - to compare their audits to other anonymous results,
- provide scientific partners and Qualixo with a large amount of data that can be used to evaluate their work on quality models.

# 3 Detailed Description

Software quality measurement is the key to software quality improvement. New trends in software development – outsourcing, merge of information systems, maintenance of "new" legacy systems in Java/C++ as well as a general awareness of the high costs of maintenance– have reinforced the need for software quality measurement. However, quality measurement is still not applied in a systematic way because of the following factors: it lacks standardization, it is difficult to represent into business terms and added values, and lacks proof of its profitability.

Most analysis tools provide low-level software metrics, which are difficult to understand for non-technical people. Software metrics target methods, classes, or packages of a software. Such metrics are syntax-driven (style checking), structural (cyclomatic complexity, etc.), object-oriented (depth of inheritance, etc.), architectural (layered model), model-based (coupling, etc.), test-driven (code coverage, etc.), or generic (performance, security, etc.).

In the Qualixo model, metrics are combined to compute marks for practice between 0 and 3. Practices are then combined to quantify - for the whole or parts of software, quality criterias and in turn quality factors such as maintainability, evolutivity, or reusability. Squale aims at reporting quality on specific dashboards and views, each tailored for a specific user: programmer, project manager, maintenance manager, sale manager, team leader, quality specialist. Software quality has to be monitored throughout the software life-cycle, in order to measure and follow progression or deterioration of quality.

The technical-economical aspect of the project targets the assessment of refactoring costs as well as the benefits of a quality-driven approach. First a technical-economical model should provide a cost for creation and modification of components based on software metrics. Then the cost of refactoring can be predicted following a quality diagnosis. Second the Squale project aims at computing the profits of a quality-driven approach, *i.e.*, the costs of non-quality. The goal is to assess the benefits of quality measurement and also to find its limits.

The main deliverables of the Squale project will be:

- an improved quality model (based on 4 levels: raw metrics, practices, criterias and factors),
- a technical-economical model (aiming at computing an estimated return of investment for quality efforts),
- a software (Squale application) that implements those models and provides a high featured UI,
- a centralized database gathering anonymous audit results from voluntary users,
- all of this widely spread under an OSI/FSF approved license, and supported by an active community.

One of the innovations of Squale lays in assessing economical factors (including the ROI) based on software quality. This is a perspective that differs from what is promoted by COCOMO, a model for estimating effort, cost, and schedule for software projects. COCOMO consists of a hierarchy of three increasingly detailed and accurate form. Assessment made by COCOMO depends on the number of lines of code and the estimated complexity of the system to build. Squale will rather use the intrinsic software quality to build its prediction.

In its early stage, Squale will target applications written in the Java and C++ programming languages. However, the model and the tool will be adaptable to other languages (e.g. Cobol, ABAP, etc.) through the development of connectors to specific tools and the parameterization of the practice-metric model. The goal is to provide homogeneous dashboards for any software pool.

We identified two crucial points that may constitute a risk from making Squale a success:

- Expressing the notion of quality without relying on discrete measurements (a typical example is cutting a long and complex function into three shorter but complex sub-functions should not increase the overall quality).

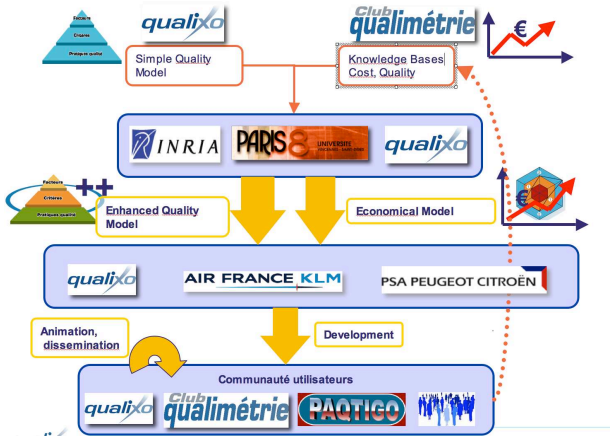


Figure 1: Structure of the Squale Project

- The huge amount of data that will have to be analyzed will probably raise some scalability issues, not only from the hardware, but also on the high level view being extracted.

# 4 Project Participants

The project is structured around three kinds of partners: two scientific partners (INRIA and LIASD), two SMEs experts in software analysis and development, two large companies (PSA and Air France-KLM). Figure 1 describes the interactions between the participants which are detailed below.

Here is the list of the participants and the main actors.

- RMoD Team of INRIA Lille Nord-Europe: Alexandre Bergel, Simon Denier, Stéphane Ducasse, Jannik Laval.
- LIASD University Paris 8: Françoise Balmas, Karine Mordal-Manet, Harald Wertz.
- Air France-KLM DSI: Arnaud Poivre and Laurent Bouhier. Air France-KLM was a co-developer of the Qualixo quality model during the last two years.
- PSA Peugeot Citroën: Thierry Bey. PSA is one of the early adopters of the Qualixo model.
- Qualixo: Fabrice Bellingard, Philippe Vaillergues and Florent Zara. Qualixo is the lead of the Squale project.
- Paqtigo: Amin Popote and Hervé Crespel.

Squale is coordinated by Qualixo. The research activity is essentially led by the academic partners (LIASD and RMoD), whereas the realization and evaluation of the produced approaches will mainly be conducted by Air France-KLM DSI, PSA Peugeot Citroën, Qualixo and Paqtigo.

# 5 Achievements and Work in Progress

According to Marinescu and Ratiu [7], the Qualixo model may be classified as a Factor-Criteria-Metrics quality model. Such a model is structured in a tree-like manner. The upper branches hold important high-level quality factors of software products. Each quality factor is composed of lower-level criteria, such as modularity and data commonality. The criteria are easier to understand and measure than the factors; thus, actual measures (metrics) are proposed for the criteria.

This model is being applied in large companies such as Air France-KLM and PSA. It uses measurements to assess software quality. These measurements cover a number of different aspects of a software, including specification accuracy, programming rules, and test coverage.

While the project is still young, several promising works are on progress.

- Squale - the platform implementing the Qualixo Model - will be released officially as open-source software early 2009. A major effort to document and clean it is taking place.
- The Qualixo model has been originally implemented on top of the Java platform. An implementation of this model, named MoQam (Moose Quality Assessment Model), is under development in the Moose open-source and free reengineering environment. The objective is to make the Qualixo model benefit from the various tools available on Moose (including the visualization engine). A first experiment has

been conducted [6]. Exporters from Moose to the Squale software are under development.

- We are assessing the metrics and practices used originally in the Qualixo model. We are also compiling a number of metrics for cohesion and coupling assessment. We want to assess for each of these metrics their relevance in a software quality setting.

- Dependency Structure Matrix (DSM), developed in the context of process optimization, is a visual approach which has been successfully applied to identify software dependencies among packages and subsystems. A number of algorithms help organizing the matrix in a form that reflects the architecture and highlights patterns and problematic dependencies between subsystems. We aim at using this matrix visualization to assess the complexity of a system and to select subsystems for quality improvment. We distinguish independent cycles and stress cycles using coloring information.This work has been implemented on top of the *Moose* open-source reengineering environment and the Mondrian visualization framework. It has been applied to non-trivial case studies such as the *Morphic UI* framework available in open-source Smalltalk *Squeak* and *Pharo*. Results have been implemented in the *Pharo* programming environment. A first experiment has already been conducted [2].

Qualixo is playing an important role in Software Quality and they are active in organizing regular workshops on this theme. Building a community around software quality is one of the major goal of Squale.

# References


[1] Jagdish Bansiya and Carl Davis. A hierarchical model for object-oriented design quality assessment. *IEEE Transactions on Software Engineering*, 28(1):4–17, January 2002.

[2] Alexandre Bergel, Stéphane Ducasse, Jannik Laval, and Romain Peirs. Enhanced dependency structure matrix for Moose. In *FAMOOSr, 2nd Workshop on FAMIX and Moose in Reengineering*, 2008.

[3] Tom Gilb. *Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage*. Butterworth-Heinemann, Newton, MA, USA, 2005.

[4] Robert L. Glass. *Building Quality Software*. Prentice-Hall, 1997.

[5] Ho-Won Jung, Seung-Gweon Kim, and Chang-Shin Chung. Measuring software product quality: A survey of ISO/IEC 9126. *IEEE Softw.*, 21(5):88–92, 2004.

[6] Jannik Laval, Alexandre Bergel, and Stéphane Ducasse. Assessing the quality of your software with MoQam. In *FAMOOSr, 2nd Workshop on FAMIX and Moose in Reengineering*, 2008.

[7] Radu Marinescu and Daniel Raţiu. Quantifying the quality of object-oriented design: the factor-strategy model. In *Proceedings 11th Working Conference on Reverse Engineering (WCRE'04)*, pages 192–201, Los Alamitos CA, 2004. IEEE Computer Society Press.

[8] Diomidis Spinellis. *Code Reading The Open Source Perspective*. Addison-Wesley, 2003.